Test Axioms – An Introduction

Paul Gerrard
Principal, Gerrard Consulting
PO Box 347, Maidenhead, Berkshire, SL6 2GU, UK
paul at gerrardconsulting dot com

Abstract

Is it possible to define a set of axioms that provide a framework for software testing that all the variations of test approach currently being advocated align with or obey? In this respect, an axiom would be an uncontested principle; something self-evidently and so obviously true and not requiring proof. What would such test axioms look like? This paper summarises some preliminary work on defining a set of Test Axioms. Some applications of the axioms that would appear useful are suggested for future development. It is also suggested the work of practitioners and researchers is on very shaky ground unless we refine and agree these Axioms. This is a work in progress.

1 DO TEST AXIOMS EXIST?

1.1 No Single Set of Guiding Test Principles

It is arguable how long the discipline we call software testing has existed, but published papers on software testing and references to testing as an activity separate from development appear in the mid-1970s. Of course, testing as an activity existed long before then and it has been suggested [1] that Ada Lovelace, by being the first programmer was, implicitly, the first tester too.

Almost every book on testing, self-promoted schools, ad-hoc and organised testing groups, and ‘test evangelists’ (let’s call them) set out their guiding principles before presenting their approach, method, dogma, techniques, heuristics etc. It seems to be in our genes as testers that we need a guiding set of principles to define our credo. Perhaps, as practitioners, we are so used to having to defend our position that these principles help our credibility and/or confidence. But it also appears that few of the books actually describe the thought process – from stated principle to advised practice.

There is a very diverse set of guiding principles being promoted in the literature. As I look at my bookshelf, I flick through early chapters of a few select books. In roughly alphabetical order, Beizer [2], Black [3], Craig and Jaskiel [4], Gerrard and Thompson [5], Hetzel [6], Kaner Bach and Petticord [7], Kaner Falk and Nguyen [8], Kit [9], Patton [10], Perry [11], Pol Teunnissen and van Veenendaal [12] all, to varying degrees present:

- A definition of testing (or several definitions, with their preferred variation)
- Some fundamental principles of testing they subscribe to
- An approach, ethos, philosophy, method that they adhere to.

Most other books show the same pattern. What do we observe here?

Firstly, we get a wide range of objectives, all of which are credible, have value and can be used as a guide. These objectives don’t reflect different agendas of the authors, but they do probably reflect the varying backgrounds and experiences of the authors and the time the
books were written. Approaches in books, papers and methodologies span a wide spectrum of high structure, high ceremony to more agile, dynamic, exploratory approaches.

Earlier writings on software testing focus on the rather narrow objective of finding bugs. Over time, the focus has broadened to cover reviews, testing as a whole-lifecycle process and the business aspects of test, namely information provision for decision-support with risk and benefits/results management as the drivers.

Second, some principles appear again and again: testing is an intellectually difficult activity; complete testing is impossible; independence of mind has value; focusing on bugs is good; testing builds confidence and so on. But some practices ‘derived from principle’ are not universally accepted. Pre-meditated, thoroughly documented, planned, prepared testing is advocated as the professional approach by some, but criticised as expensive, ineffective, stupifying and inflexible by others. Dynamic, concurrent test design and execution, using heuristics and an agile mentality are promoted by some, but grudgingly acknowledged as being of limited use in small projects by others.

The differing approaches advocated by the various authors may reflect different backgrounds and experiences. Perhaps this is the reason why approaches are promoted and supported so assertively. If an approach is based on one's experience, it is hard to compromise, as one's experience cannot be changed. Obviously, an approach that is appropriate to a web start-up is probably not appropriate for a safety-critical application, or a compliance or evidence-driven testing project. Context is everything.

But why do different authors promote different basic principles?

1.2 The Foundations of Software Testing are Disputed (to say the least)

Is it realistic to believe that there are an underlying set of principles that underpin all testing, regardless of context? Neil Thompson [13] attempted to build a bridge between the various ‘schools’, identify a set of ‘always-good practices’ and used the Goldratt thinking tools. The context-driven school resist the notion of 'Best Practices'. One might quibble with the context-driven principles as stated [14], but it must surely be acknowledged that all testing and practices are context-dependent and there can be no 'best practices' for all contexts.

A better characterisation of the two schools might be those that promote test design as a pre-meditated activity or one that is contemporaneous to test execution. But the split between the schools is not clear-cut – it is one of emphasis, rather than slavish adherence.

The Software Testing discipline seems not to have an agreed foundation. An unsafe, unsatisfactory and indefensible situation!

1.3 The Contexts in Which Test Axioms Apply

There is a valid objection to the notion of axioms in testing. The business spectrum of contexts in which projects exists is huge. The technical spectrum is just as wide. How can there be a set of ‘laws’ that describe or define the approaches testers must make? Testing has been described as a ‘social science’ [15]. How can there be a set of immutable laws for a human, error-prone activity like that?

In physics, Newton’s laws were shown to be an approximation when Einstein properly accounted for relativistic effects. As time passes, every law seems to be shown to be approximately true, or true in only some contexts. In the context of non-relativistic motion (i.e. velocities that are within our normal human experience) Newton’s laws apply with acceptable accuracy but they are an approximation.

Inevitably, there cannot be a set of Test Axioms which hold for all contexts, so let me say this: The ‘Testing Axioms’ postulated herein are axiomatic in conventional projects.
1.4 The Test Axioms Apply in Conventional Projects

If an axiom (stated here) does not hold in your context, then your context is ‘eccentric’. What do I mean by eccentric? Here are some examples (more than one of which I have experienced personally):

- You are asked to test an object or system that does not exist;
- The outcome of testing is of no interest to anyone on the project;
- There are no limits in terms of time, cost or effort in your project;
- Testing is regarded as an activity with no outputs or deliverables (of value);
- Testing is regarded as a purely clerical activity;
- Testers are required to lie about or suppress their findings.

In my experience, projects that exhibit these characteristics could reasonably be described as, ‘Projects from Hell’ – at least from a tester’s perspective.

1.5 So, What Should Test Axioms Look Like?

If the industry needs an agreed set of underlying axioms, what would they look like? Here are my suggested criteria for Test Axioms:

- From the perspective of any software tester, they are self-evidently true.
- The axioms apply to any test approach from an end to end perspective to the perspective of an individual doing just a little testing.
- The axioms are distinct from guidelines or principles that reflect a particular context. They are context insensitive.
- They are not practices, although ‘established’ or ‘novel’ practices may be chosen to adhere to or implement the axiom.
- A testing approach must adhere to or implement the axioms or be deemed 'incomplete'. Different approaches reflect a difference in emphasis across the range of axioms, rather than a different set of implemented axioms.

The axioms represent mechanisms designed to meet the objectives of the testing in scope. A mechanism may be a well-defined, documented process, an informal or even ad-hoc activity - but that mechanism must be understood and used by participants in the test.

2 THE PROPOSED TEST AXIOMS

Table 1 presents the tabulated set of sixteen proposed Test Axioms. Each Test Axiom has a name. This is just shorthand that makes cross-referencing of the Axioms easy. The Stakeholder Axiom is an example. The Axioms are most commonly set out in a matter-of-fact way, which is what I propose they are.

The implications of an Axiom are set out in a descriptive way, as an Action or Narrative. To better explain an Axiom, the consequences of disregarding it are set out in the ‘if you don’t recognise the axiom’ column.
### Table 1, Proposed Test Axioms

<table>
<thead>
<tr>
<th>Name</th>
<th>Axiom</th>
<th>Action/Narrative</th>
<th>If you don’t recognise the axiom</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stakeholder</td>
<td><strong>Testing needs stakeholders</strong></td>
<td>Identify and engage the people or organisations that will use and benefit from the test evidence you are to provide.</td>
<td>You won’t have a mandate or any authority for testing. Reports of passes, fails or enquiries have no audience.</td>
</tr>
<tr>
<td>Test Basis</td>
<td><strong>Test needs a source of knowledge to select things to test</strong></td>
<td>Identify and agree the goals, requirements, heuristics, risks, hearsay needed to identify targets of testing.</td>
<td>How will you select things to test?</td>
</tr>
<tr>
<td>Test Oracle</td>
<td><strong>Test needs a source of knowledge to evaluate actual behaviour</strong></td>
<td>Define the sources of knowledge whether documented, physical, experience or hearsay-based to be used to determine expected behaviour.</td>
<td>How will you assess whether tested software behaves correctly or not?</td>
</tr>
<tr>
<td>Fallibility</td>
<td><strong>Your sources of knowledge are fallible and incomplete</strong></td>
<td>Test bases, oracles, requirements, goals are fallible because the people who write them are human.</td>
<td>It is naive to think otherwise, as human error has an impact at every stage of the development lifecycle.</td>
</tr>
<tr>
<td>Scope Management</td>
<td><strong>If you don’t manage scope, you may never meet stakeholder expectations</strong></td>
<td>You must have a mechanism for identifying and agreeing the items in and out of scope (documentation, software or other deliverable or output) and managing change.</td>
<td>It is possible, and probable that stakeholders will assume you will test ‘everything’. You may also test and report progress of tests that are of no interest to stakeholders.</td>
</tr>
<tr>
<td>Design</td>
<td><strong>Test design is based on models</strong></td>
<td>Identify, adopt and agree a model or models to be used to select test cases.</td>
<td>Test design will be subjective, random and inconsistent – and not be credible.</td>
</tr>
<tr>
<td>Coverage</td>
<td><strong>Testing requires a coverage model or models</strong></td>
<td>You must have a means of evaluating narratively, qualitatively or quantitatively the testing you plan to do or have done.</td>
<td>You may not be able to answer questions such as, ‘what has been tested?’, ‘what has not been tested?’, ‘have you finished testing?’</td>
</tr>
<tr>
<td>Delivery</td>
<td><strong>The usefulness of the intelligence produced by test determines the value of testing</strong></td>
<td>Define what and how you need to report from testing. Define a mechanism, frequency, media and format for the evidence to be provided.</td>
<td>Different stakeholders require different formats and analyses of intelligence and may not find your test reporting useful for decision making.</td>
</tr>
<tr>
<td>Environment</td>
<td><strong>Test execution requires a known, controlled environment</strong></td>
<td>Establish the need and requirements for an environment to be used for testing, including a mechanism for managing changes to that environment – in good time.</td>
<td>Environments may be delivered late or not at all or not be as required. This will delay testing or undermine the credibility of testing performed.</td>
</tr>
</tbody>
</table>
3 APPLICATIONS

There appear to be several potential applications of the Test Axioms:

- the need for a practice in context can be justified;
- as drivers for questions in a test approach assessment or process audit;
- as a thinking tool to support stakeholder engagement and test strategy;
- as a framework for tester education and development.

In short, the value of each Axiom is primarily as a Thinking Tool for testers. Some are most appropriate to test strategy and management, but they can also apply to the very next test you need to plan, create and execute as a hands-on tester.
4 CHALLENGE TO ACADEMIA AND INDUSTRY

This paper has suggested that the founding principles of all software testing are undefined, disputed and of varying value. A set of Testing Axioms upon which all of our testing related activity and research can be founded has been proposed. If the industry cannot agree on such a set of Axioms, how can we talk or work with confidence in our profession?

This is a work in progress and the author will gratefully receive comments, criticisms or suggestions for further Test Axioms.

5 REFERENCES

www.tiscl.com
testing.com
www.kaner.com/pdfs/KanerCUSECstss.pdf